UNIVERSITY OF
ILLINOIS LIBRARY
AT URBANA-CHAMPAIGN

CENTRAL CIRCULATION BOOKSTACKS

The person charging this material is responsible for its renewal or its return to the library from which it was borrowed on or before the **Latest Date** stamped below. **You may be charged a minimum fee of $75.00 for each lost book.**

Theft, mutilation, and underlining of books are reasons for disciplinary action and may result in dismissal from the University.

TO RENEW CALL TELEPHONE CENTER, 333-8400

UNIVERSITY OF ILLINOIS LIBRARY AT URBANA-CHAMPAIGN

FEB 2 1

AUG 7

AUG 2 2006

DEC 0 7 2010

When renewing by phone, write new due date below previous due date.

L162

I l 6r
no. 979
Math.

UIUCDCS-R-79-979

UILU-ENG 79 1728

A MANUAL OF SIMPL
A LOGIC NETWORK SIMULATOR FOR PLATO

by
Sadahiro Isoda
Elyas Farzan-Kashani

July 1979

DEPARTMENT OF COMPUTER SCIENCE
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN · URBANA, ILLINOIS

THE LIBRARY OF THE
1980
UNIVERSITY OF ILLINOIS
URBANA-CHAMPAIGN

A MANUAL OF SIMPL
A LOGIC NETWORK SIMULATOR FOR PLATO

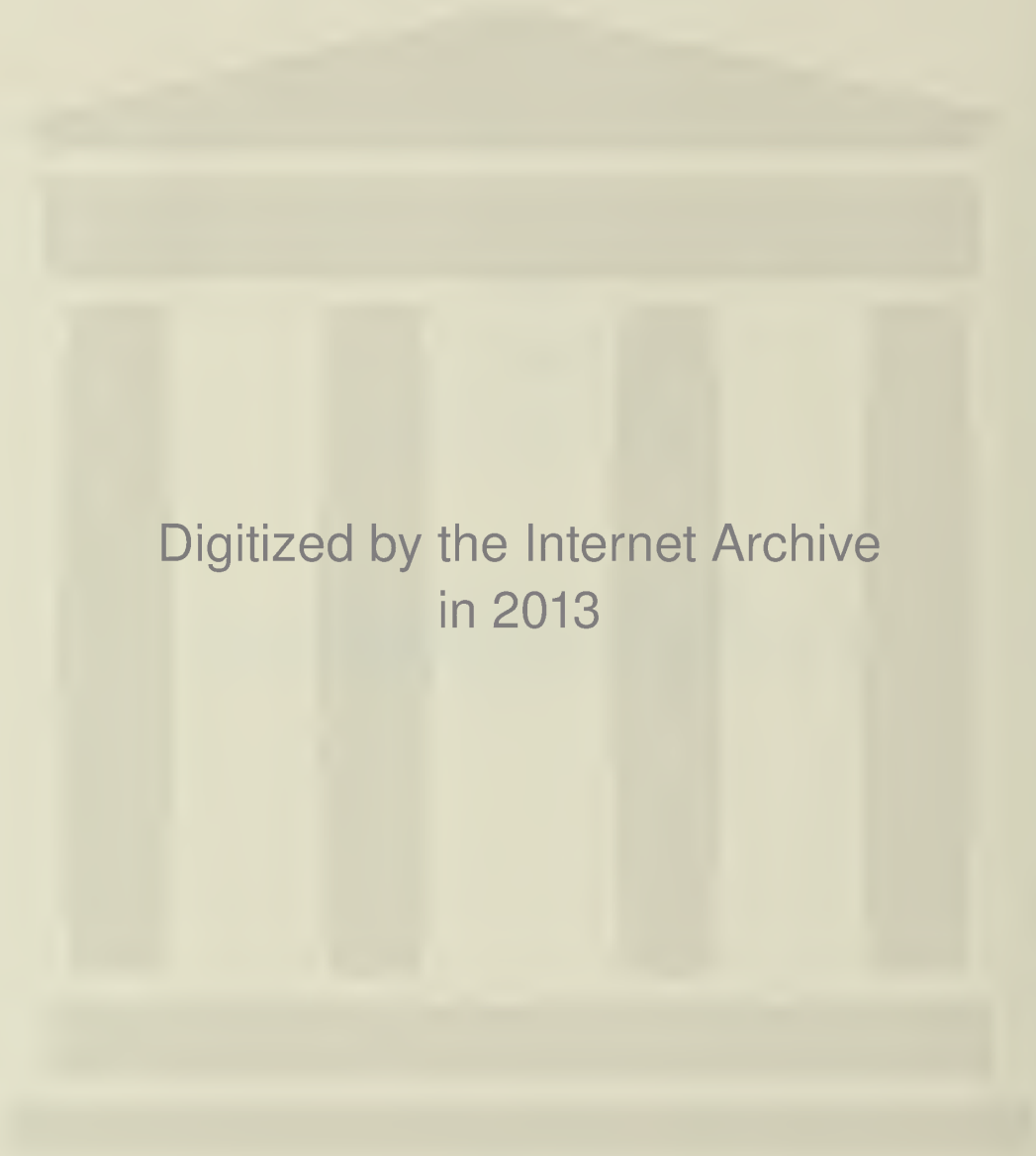
BY

SADAHIRO ISODA

ELYAS FARZAN-KASHANI

DEPARTMENT OF COMPUTER SCIENCE
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN
URBANA, ILLINOIS 61801

July 1979


Digitized by the Internet Archive
in 2013

http://archive.org/details/manualofsimpllog979isod

Introduction

This is a user's manual of SIMPL, a logic network simulator for PLATO. SIMPL is a program which can interactively simulate logic networks, based on the computer-based education system, PLATO.

SIMPL has the following facilities:

i) Constructing any network consisting of gates (AND, OR, NOT, NAND, NOR, or XOR) and input terminals.

ii) Supplying input value sequences to input terminals.

iii) Assigning initial values to gates.

iv) Showing timing charts for specified gates and input terminals.

SIMPL is quite easy to use even for a novice, because the user, following messages from SIMPL, can use it almost by means of touching the screen.

Screen

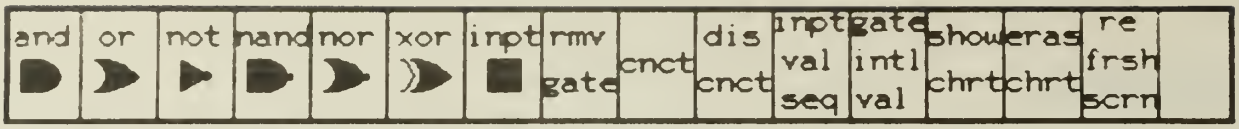
Figure 1 shows what the screen looks like initially. The screen consists of the following four parts:

i) an array of 15 commands at the top,

ii) a message area,

iii) a two-dimensional array of locations for gates or input terminals, and

iv) a blank space to be used for displaying timing charts at the bottom.



11 12 13 14 15 16 17

21 22 23 24 25 26 27

31 32 33 34 35 36 37

41 42 43 44 45 46 47

51 52 53 54 55 56 57

61 62 63 64 65 66 67

Fig. 1 The initial screen

How to place gates

In order to place a gate at a particular location, the user first touches one of the six commands on the left half of the command array. For example, he touches the command "and". Then SIMPL writes the message ""and" at which location?" just under the array of commands. The user touches the location where he wants to place the AND gate, and then an AND gate appears there.

If the user wants to place another gate of the same type (namely an AND gate) at any other location, he does not have to touch the command "and" again. He has only to touch the location where he wants to place a gate.

If he wants to place a gate of a different type from the one just placed, he has to touch the corresponding command before touching a location.

How to place input terminals

The user follows the same procedure as for placing gates, with the difference that he touches the command "inpt" (input).

How to place connections

In order to connect gates (or an input terminal and a gate), the user first touches the command "cnct" (connect). Then SIMPL displays the message "Connect from which gate or input?". The user then touches the origin (i.e., the gate or input terminal from which he wants a connection to be drawn), and SIMPL makes the gate or input terminal blink and displays the next message "From gate xx to which gate?". The user then touches the destination (i.e., the gate to which he wants a connection to be drawn), and SIMPL draws a connection.

If he wants to place another connection, he does not have to touch the

command "cnct". He has only to touch the origin and destination.

How to supply input value sequences to input terminals

The user touches the command "inpt-val-seq" (input value sequence), and then the message "Which input?" appears. He touches the input terminal to which he wants to supply an input value sequence. The message "Press 0,1,z(constant 0) or u(constant 1)." appears, and a time axis for a timing chart also appears at the bottom of the screen. If he wants to supply the sequence "0110001100011...", he presses the "0" and "1" keys in this order. Every time he presses "0" or "1", a corresponding line segment which is part of the entire timing chart is drawn on the time axis. He has to specify an input sequence 27 bits long. "z" and "u" are for level input --- i.e., if an input terminal takes a constant value of "0" or "1", the user has only to press "z" or "u" only once, respectively, instead of pressing "0" or "1" 27 times. The user can also press "z" or "u" after pressing "0" or "1" any number of times.

If he wants to supply another input value sequence to any input terminal, he does not have to touch the command "inpt-val-seq" again. He has only to touch the input terminal to which he wants to supply an input value sequence, and then press "0", "1", "z", or "u" keys.

How to assign initial values to gates

If the user wants to analyze a sequential network, he has to assign initial values to gates. First the user touches the command "gate-intl-val" (gate initial value), and then the message "Touch gates for initial value 1." appears. Then he touches those gates to which he wants to assign an initial

value of "1". The gates left untouched are given an initial value of "0".

How to show timing charts

Now the user is ready to show timing charts of the network he has just constructd. He touches the command "show-chrt" (show chart). Then the message "Which gate or input?" appears. He touches a gate or an input terminal, and then a timing chart appears at the bottom of the screen.

At most six timing charts can be shown on the screen at a time. They are arranged in a two-dimensional array with three rows and two columns, as illustrated in Fig. 11. The first chart is shown in the first row of the left column, the second in the second row of the left column, and so on. When the seventh chart is to be displayed, the first one is erased and replaced by the seventh one.

Example

If you are to analyze how a flip-flop network (Fig. 2a) performs when the J and K inputs are kept at "1" and the clock pulse duration is two or three (Fig. 2b), do the following:

1. Sign-on and start "SIMPL". The initial screen appears (Fig. 1).
2. Touch the command "nand", and then locations 23, 43, 25, and 45.
3. Touch the command "inpt", and then location 31 (Fig. 3). (The input terminal just placed is for the clock input. Any other input terminals need not be created, since J and K inputs are kept at "1" and only NAND gates are used.)
4. Touch the command "cnct", and then locations 31 and 23. A connection is now drawn between these two locations (Fig. 4).

5. Draw other connections by touching appropriate locations (Fig. 5).
6. Touch the command "inpt-val-seq", and then location 31. A time axis appears at the bottom of the screen (Fig. 6).
7. Press the clock pulse sequence "0110001100011...", to analyze the case when the clock pulse duration is two. A timing chart for the input terminal is now drawn (Fig. 7).
8. Touch the command "gate-intl-val", and then locations 23, 43, and 25.
9. Touch the command "show-chrt", and wait for a while. The message "Now calculating." is shown (Fig. 8).
10. When the calculation is finished, the message "Which gate or input?" appears. Touch locations 25 and then 45. Now two timing charts appear next to the one for the input terminal (Fig. 9).
11. Next, analyze the case when the clock pulse duration is three. Touch the command "inpt-val-seq", and then location 31. Then press "0111000111000111..." (Fig. 10).
12. Touch the command "show-chrt", and then locations 25 and 45 (Fig. 11). We now have a set of timing charts showing oscillation.
13. We can conclude that the performance of the network with J=K=1 is stable when the clock pulse duration is two but becomes unstable when the clock pulse duration is three.

Fig. 2a A J-K flip-flop

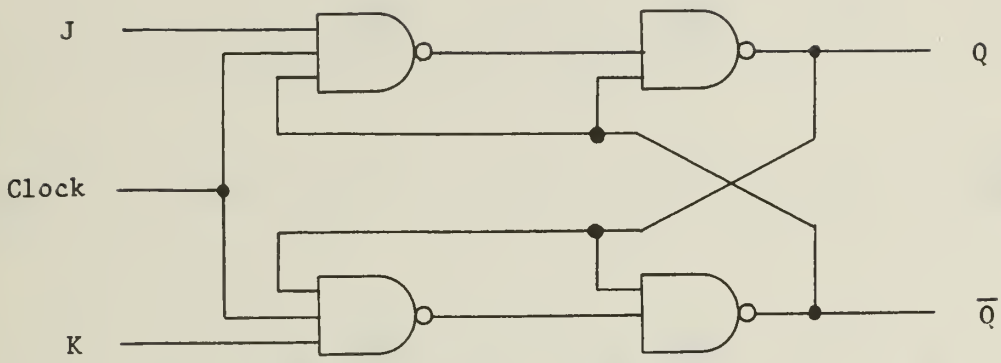


Fig. 2b Clock pulse sequences

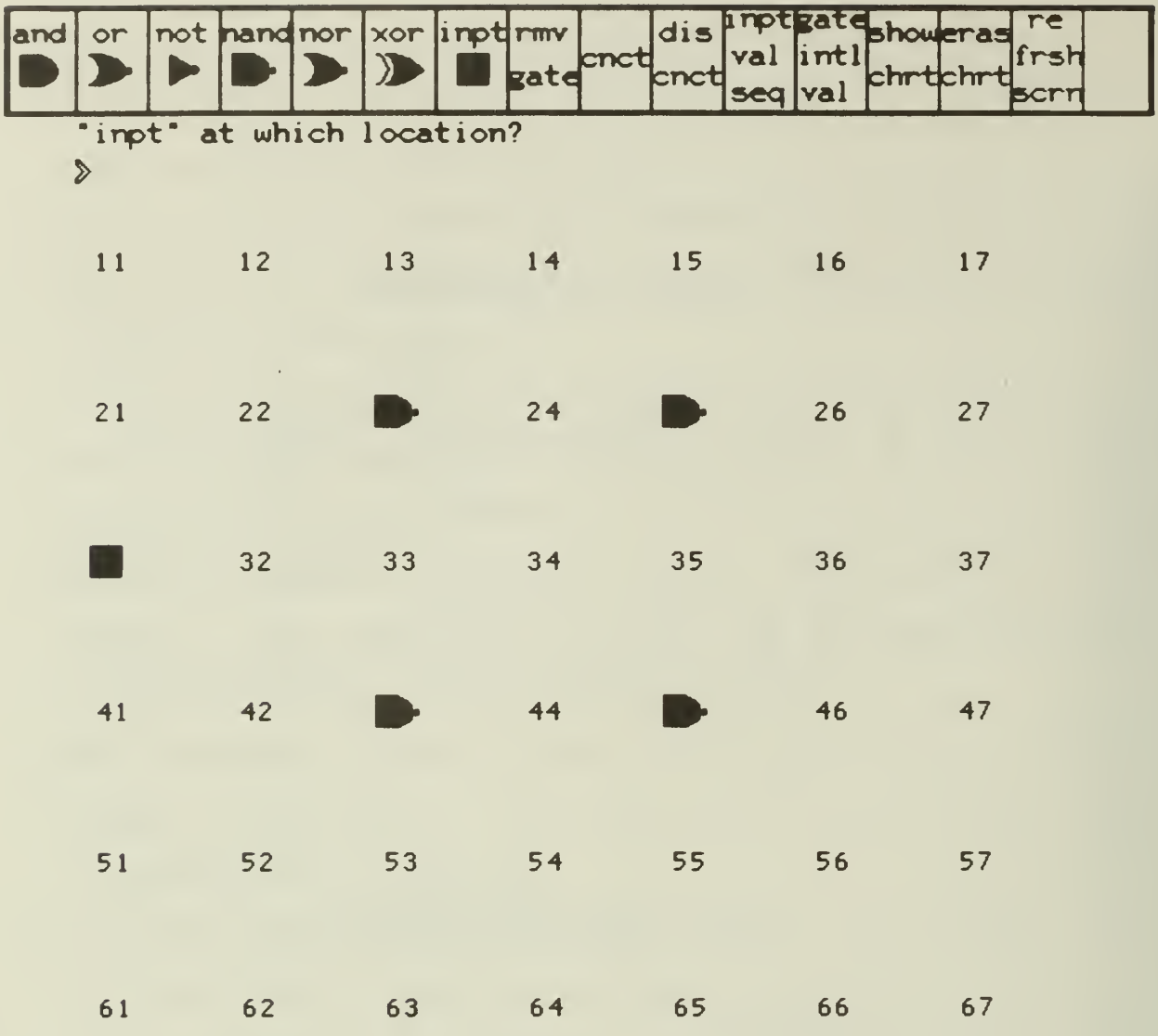


Fig. 3 After placing four NAND gates and an input terminal

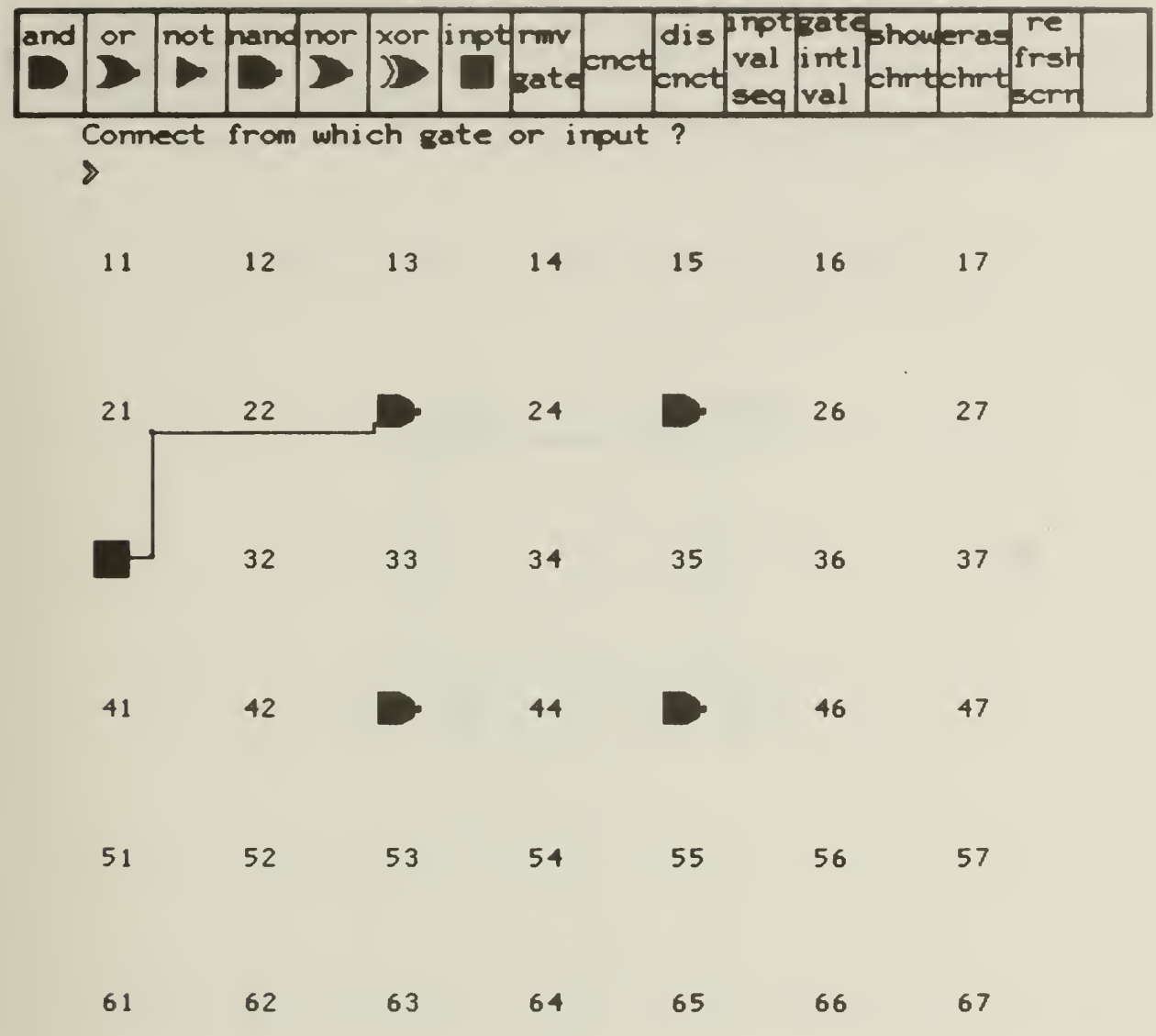


Fig. 4 After drawing the connection from the input terminal at location 31 to the NAND gate at location 23

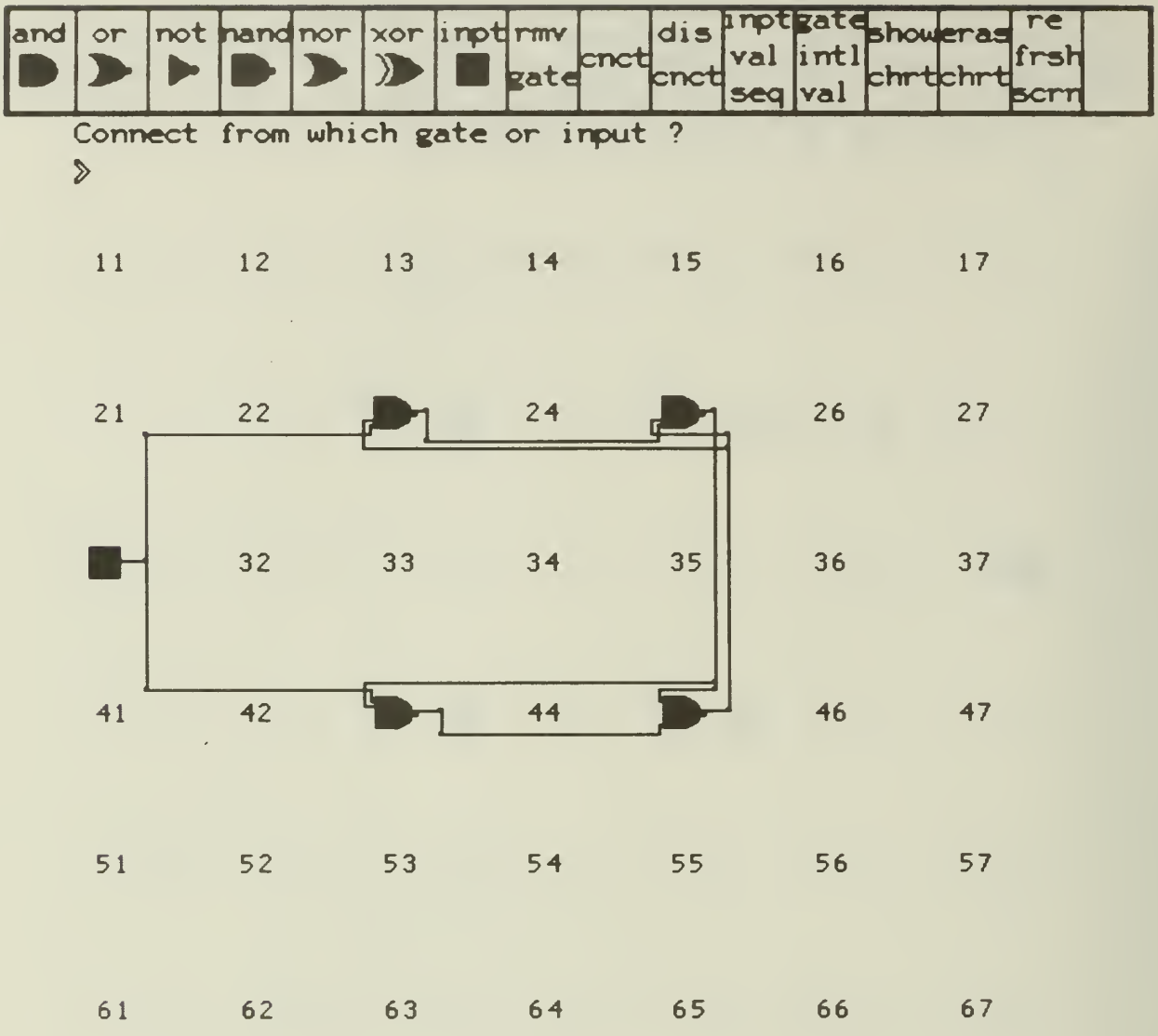


Fig. 5 After drawing all the necessary connections

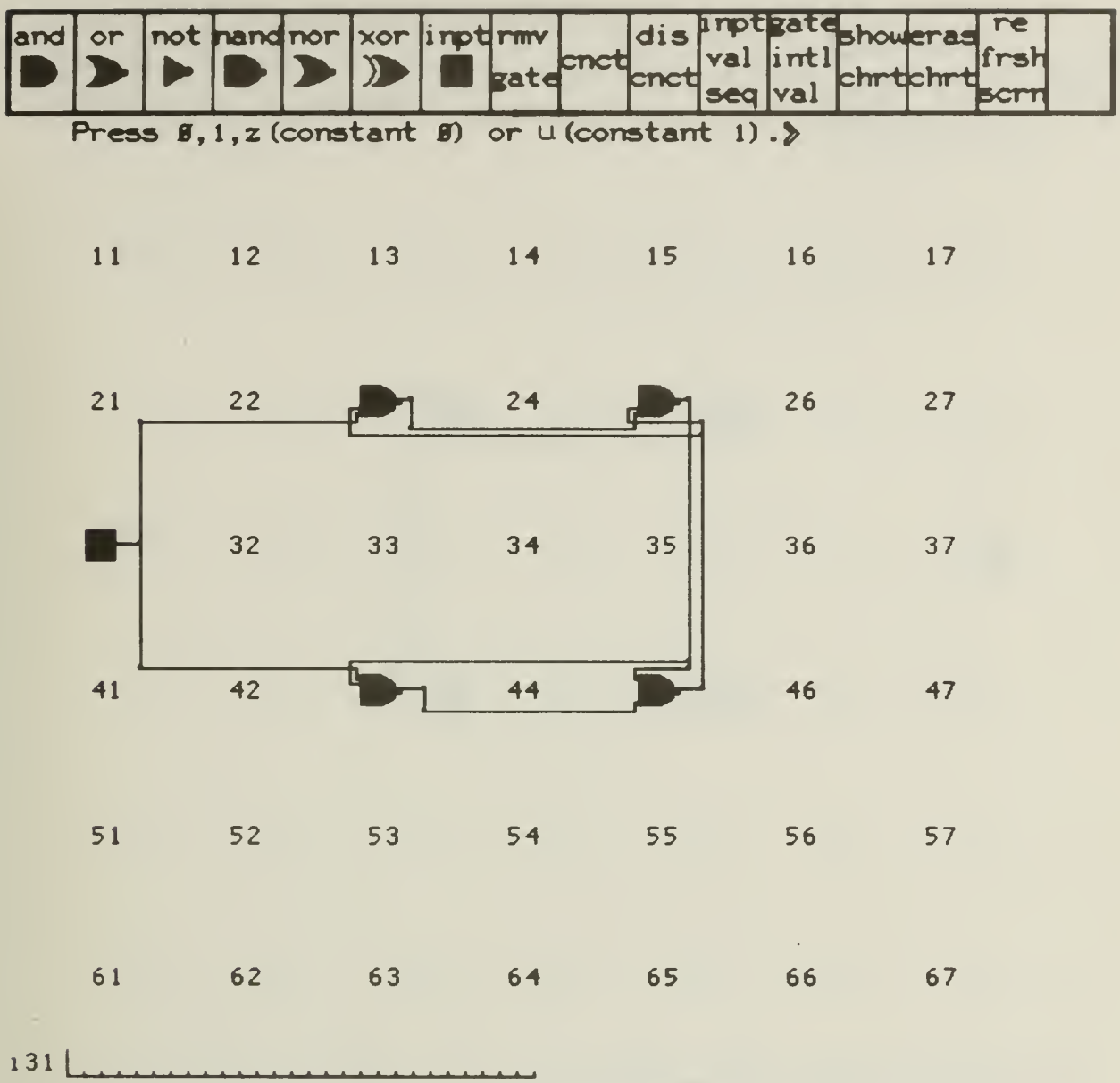


Fig. 6 After touching the command "inpt-val-seq", and then the input terminal at location 31. (A time axis i31 is now drawn.)

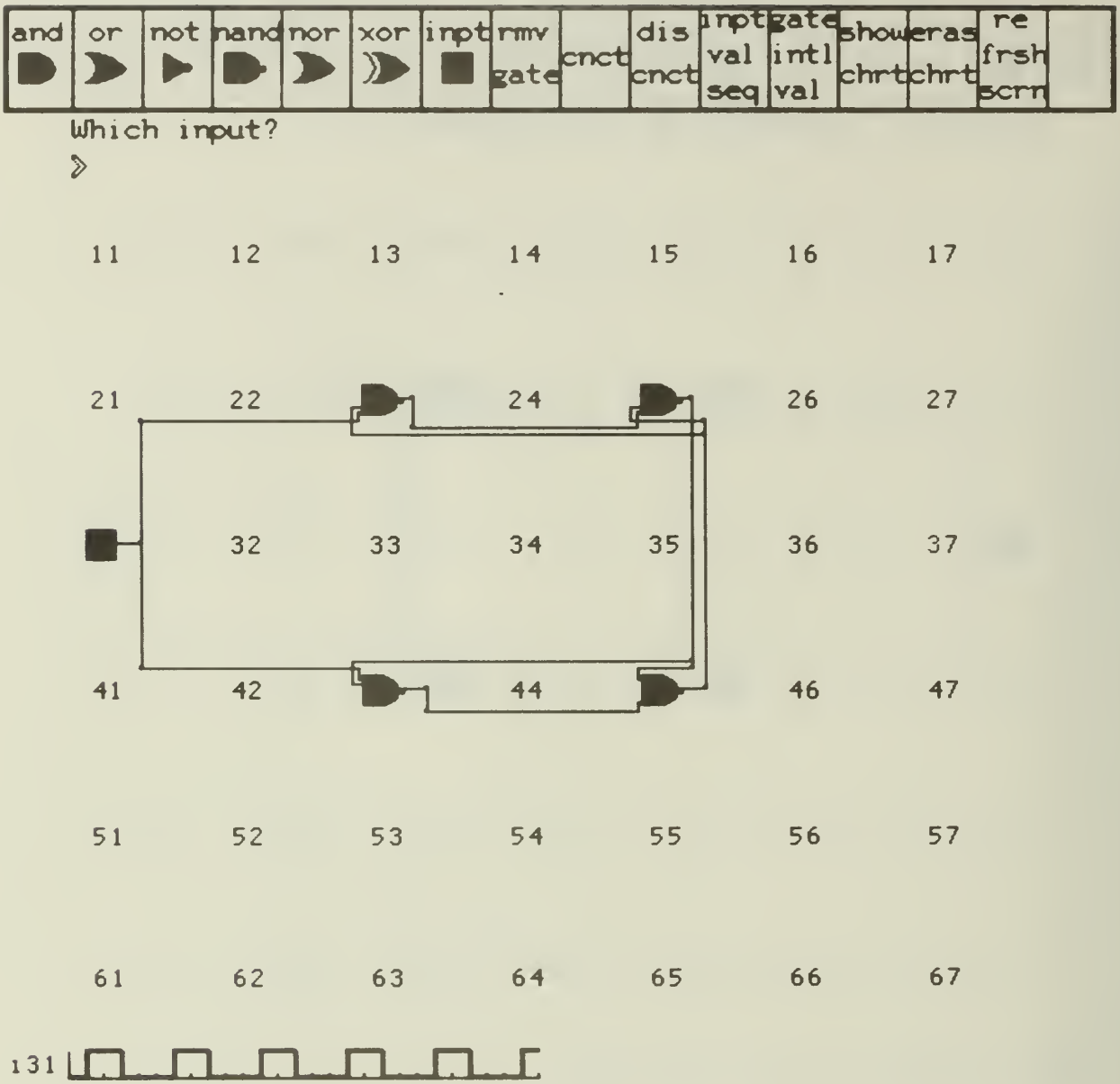


Fig. 7 After pressing "0110001100011...". (The timing chart for the clock is now drawn.)

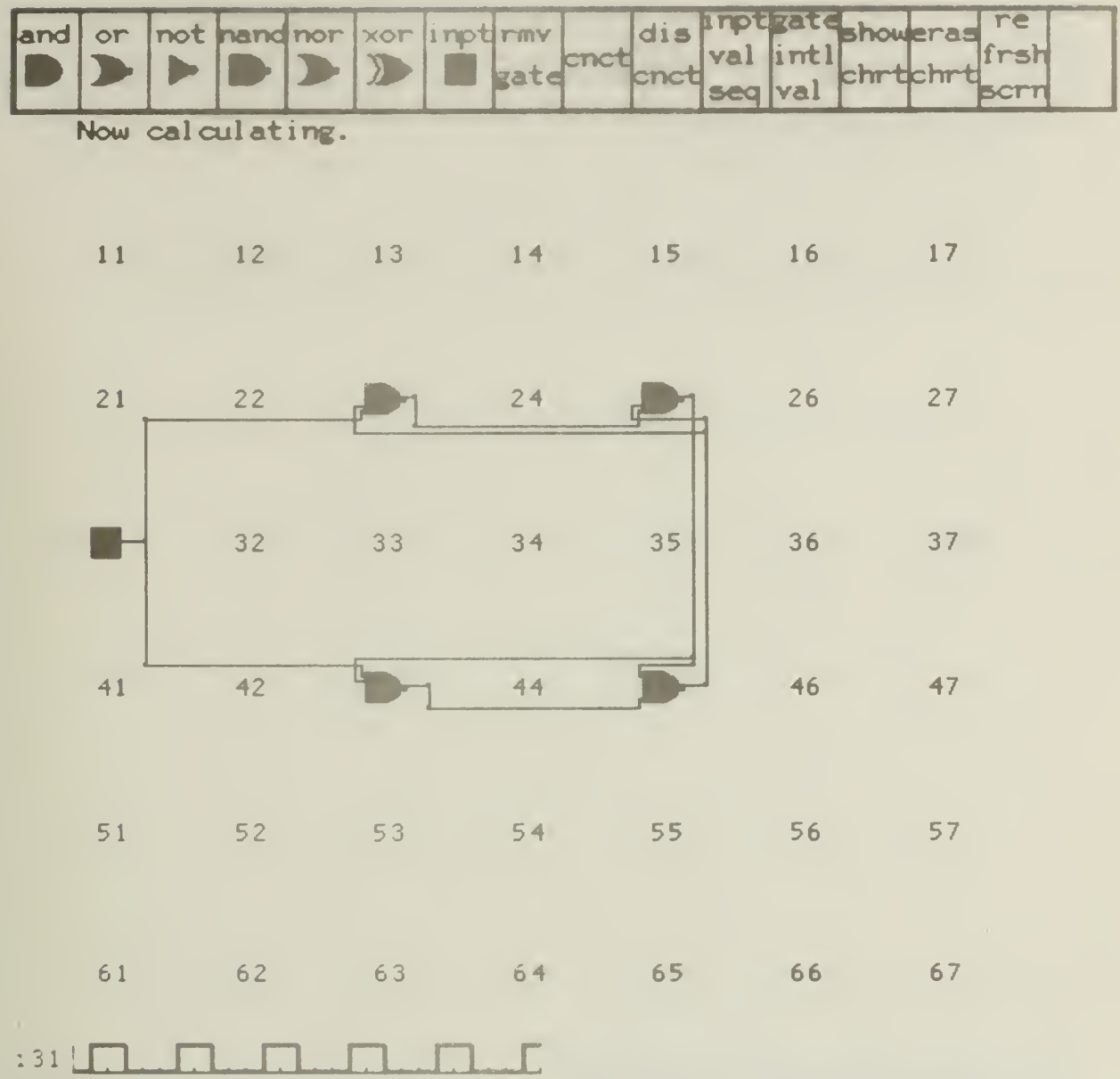


Fig. 8 After touching the command "show-chrt"

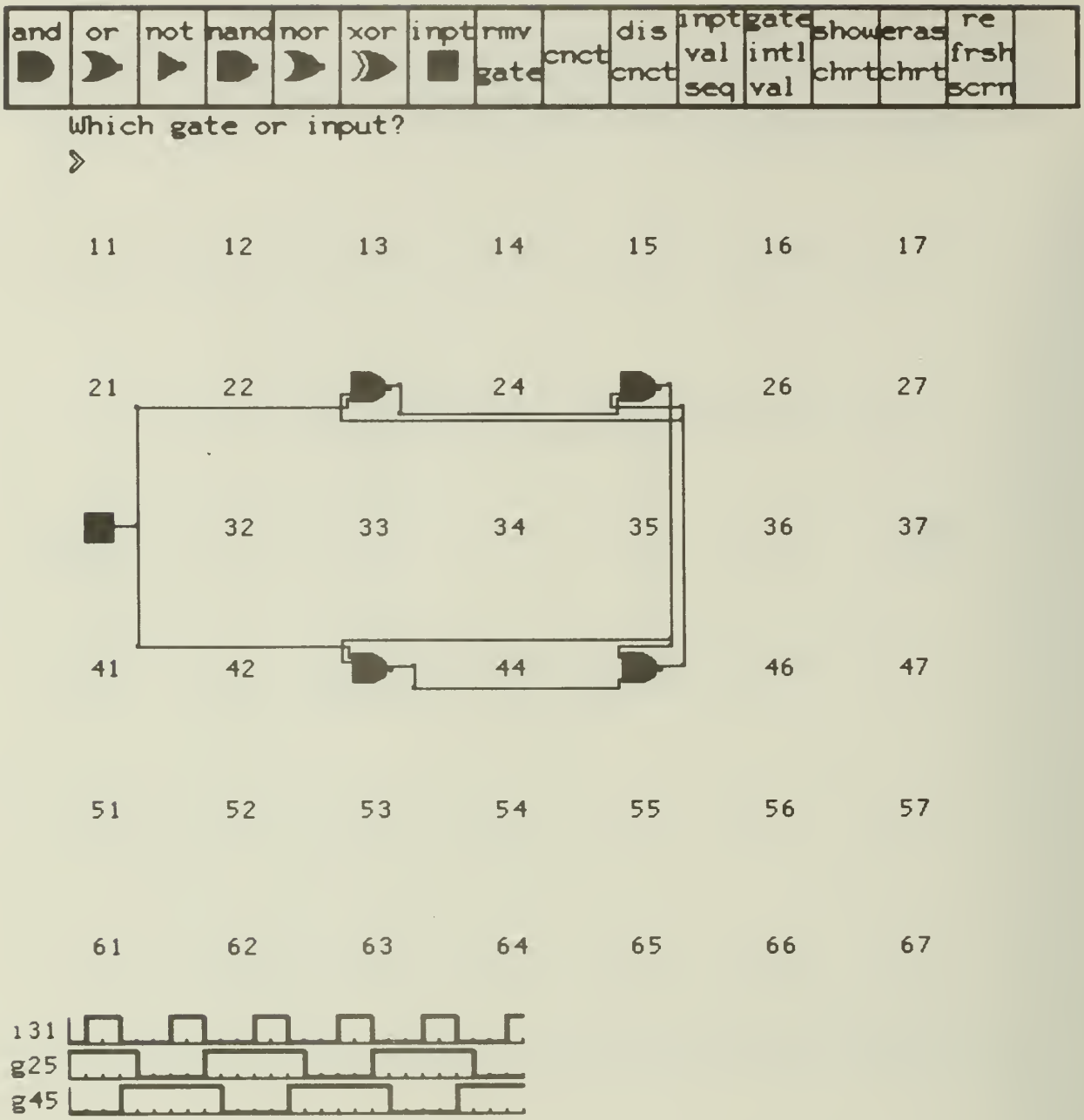


Fig. 9 After touching the gates at locations 25 and 45. (Two timing charts, g25 and g45 appear next to the chart i31.)

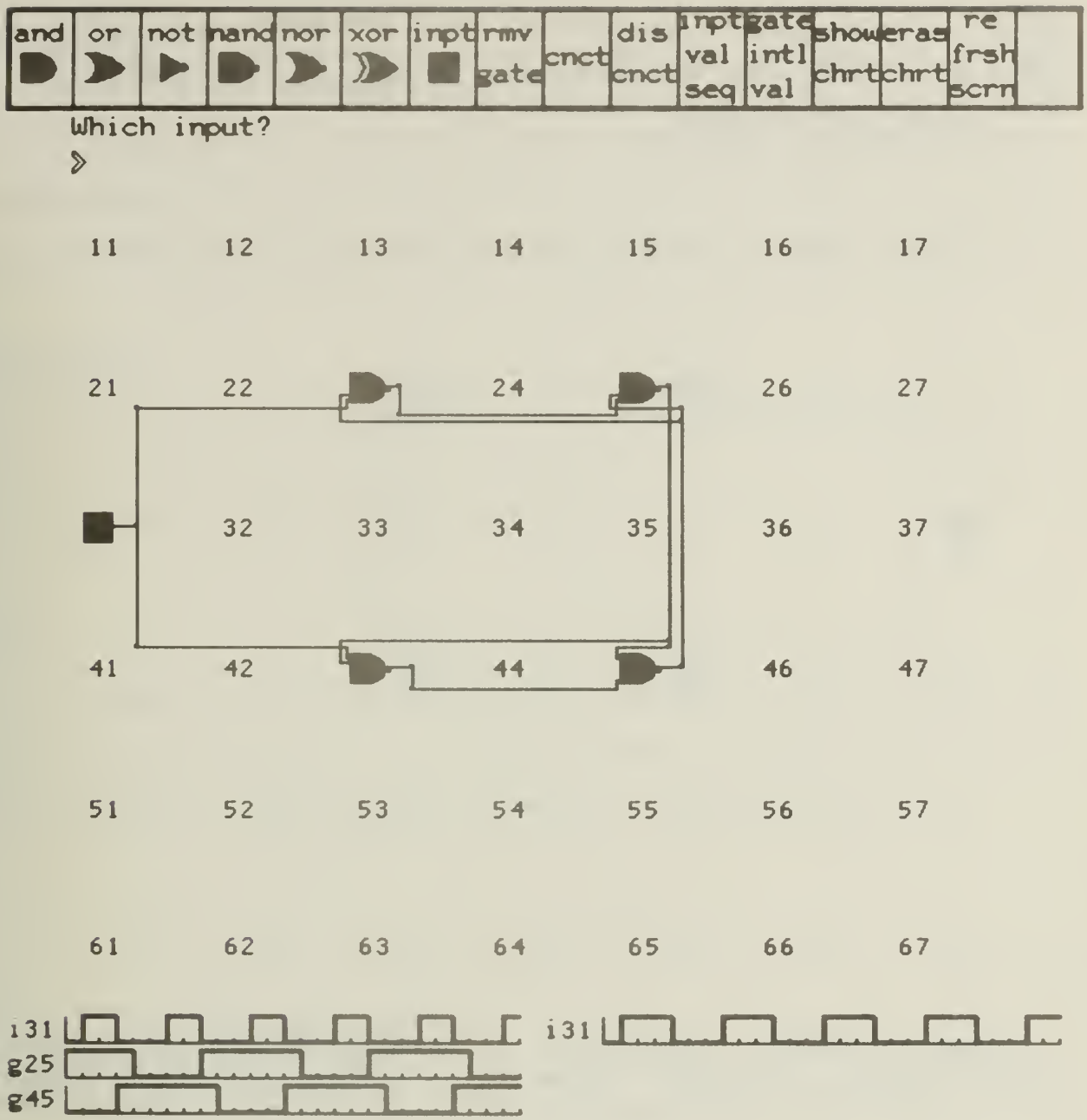


Fig. 10 After touching the command "inpt-val-seq" again, and then pressing "0111000111000111...". (A timing chart for the new clock pulse appears.)

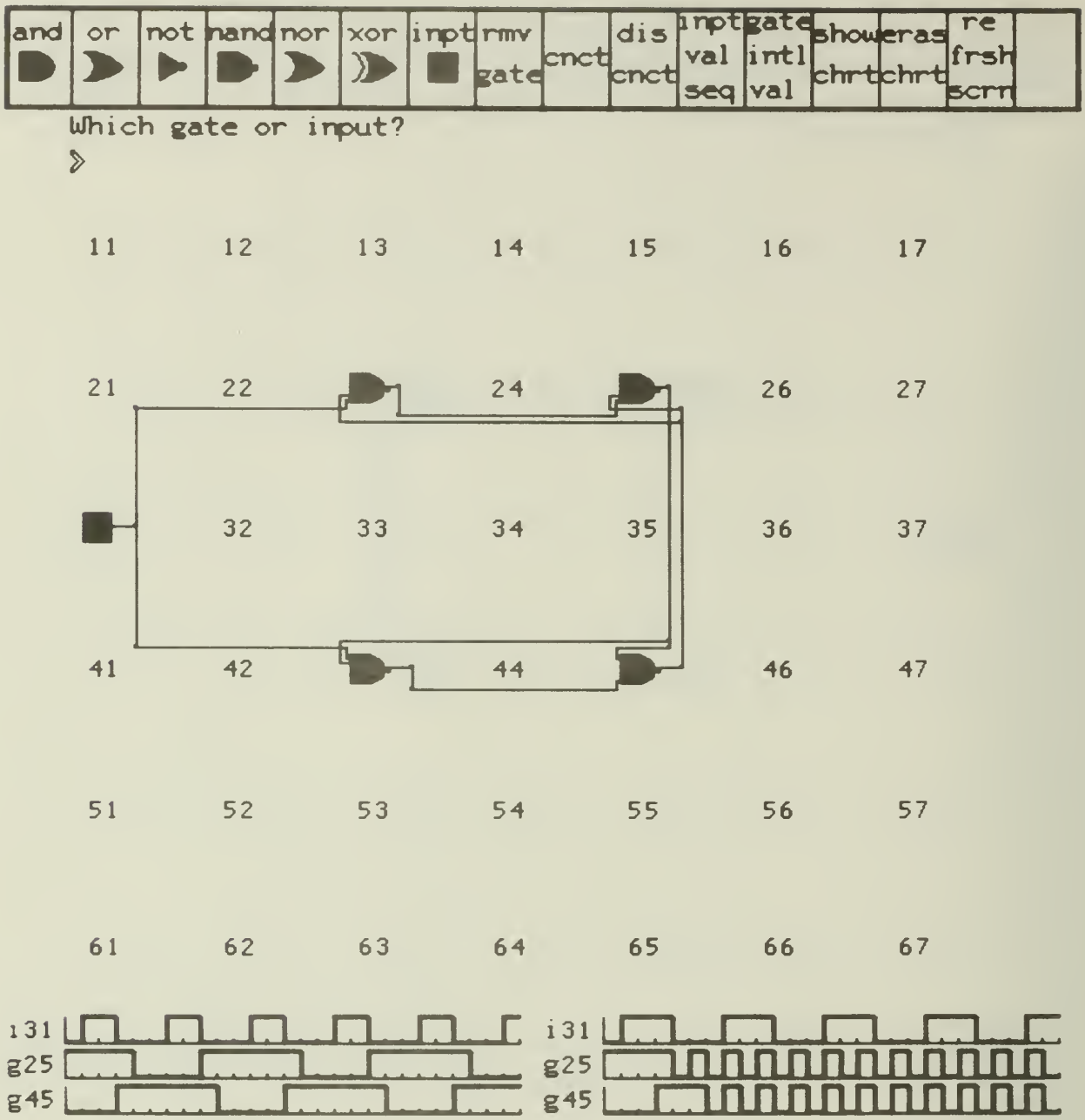


Fig. 11 After touching the command "show-chrt", and then the two gates at locations 25 and 45. (We now know the outputs of the flip-flop oscillate when the clock pulse duration is three.)

Other commands

SIMPL also has two other commands which are not explained in the above text. They are the command "eras-chrt" (erase charts) for erasing all the timing charts shown on the screen, and the command "re-frsh-scrn" (refresh screen) for erasing the whole screen and displaying the initial screen again.

Limitations

i) There can be at most 42 gates or input terminals on the screen.

ii) The maximum number of input connections to a gate, i.e., the fan-in, is six.

Note

The user must not forget to assign initial values to gates when he deals with sequential networks. If he forgets to assign an initial value of "1" to gate 25 in step 8 of the example, he will get an oscillatory result.

APPENDIX A

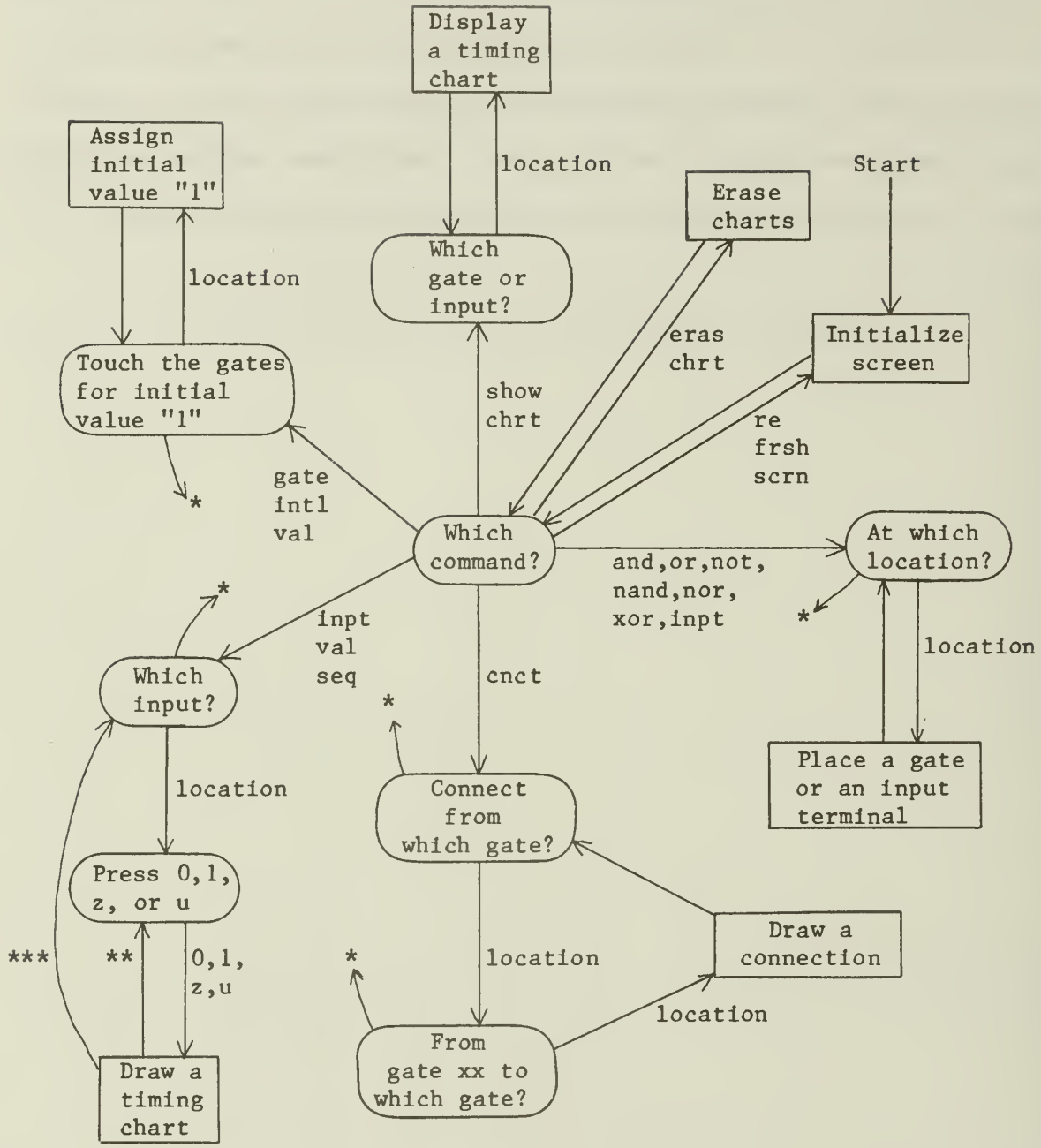
Program Description

This is a brief description of the internal structure of the program, SIMPL. It describes the dynamic structure of the program, outlines the module structure, and discusses about the usage of the student variables.

This is only an introduction to the internal structure of the program. For further information, read the program listing in APPENDEX B.

State transition diagram

The dynamic structure of SIMPL is described by the state transition diagram in Figure A.1. In this figure each round node represents a stable state in which SIMPL writes a message (shown inside the node) on the screen and then waits for inputs from the user. Each square node represents a transitory state in which SIMPL does the action (shown inside the node).



* When a command is fed, the control is transfered directly to the state corresponding to the command.
** When the timing chart is not completed.
*** When the timing chart is completed.

Fig. A.1 State Transition Diagram

Module structure

The program consists of a main routine ("main"), six major subroutines ("gates", "connect", "inpvseq", "gtinitl", "calcul", and "chart"), and some low-level subroutines (Fig. A.2).

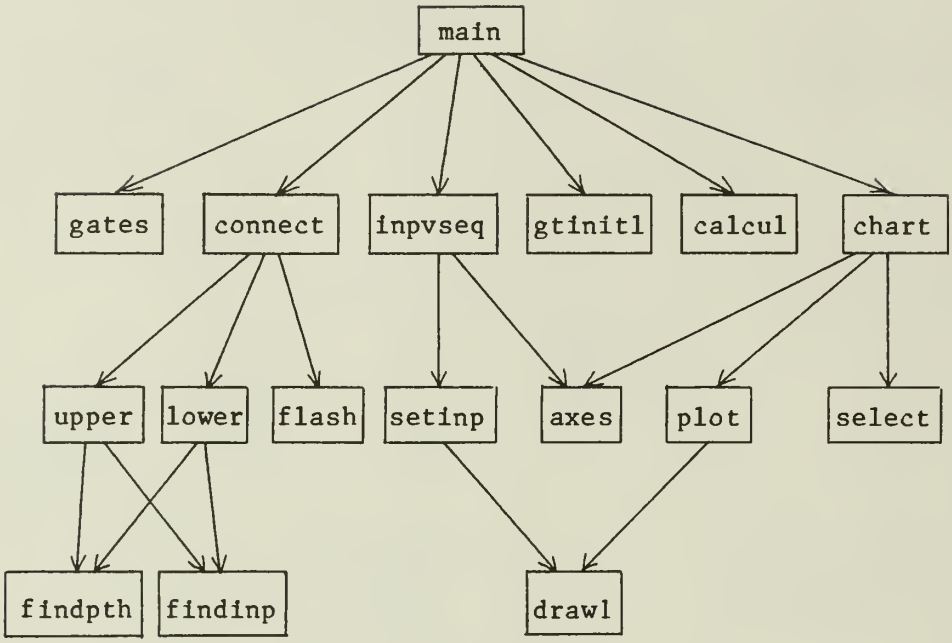


Fig. A.2 Module Structure

Usage of student variables

In the PLATO system each program can use at most 150 student variables --- namely n1 through n150. Student variables are the only resource that can be used as working storage in each program. When we develop a large program, 150 student variables may not be sufficient and we may have to use some of them for more than two purposes.

The necessity of multiple use of student variables may make the use of modular structure difficult, but we can overcome this problem by observing these two rules:

i) Variables that are local to a subroutine are defined in it, thus localizing such variables.

ii) The lowest-level subroutines use the student variables n1 through n10, the second lowest-level subroutines use n11 through n20, and so on. n31 through n150 are used as global variables.

Future developments

The following functions are to be added to SIMPL:

i) to remove gates, input terminals or connections. (This function will be convenient when the user wants to erase part of a network drawn on the screen.)

ii) to store network structure information in a file, or to retrieve it from the file. (This function will be convenient when large networks are to be simulated more than once.)

APPENDIX B

Program Listing

```
CHARTST ,GATES
IF      NOT(ZRETURN)
.       AT      1215
.       WRITE   ↑LOADING CHARACTERS, PLEASE BE PATIENT
.       CHARSET ,GATES
.       AT      1215
.       ERASE   40
ENDIF
$
DEFINE  ARRAY,ALL(150)=N1
$
DEFINE  ARRAY,NTWK(42)=N67      $$ N67 TO N108
        ARRAYSEGV,TYPE(42)=N67,1,4  $$ 1=AND,2=OR,3=NOT,4=NAND,5=NOR,6=EOR
$       BITS 5,6 ARE EMPTY
        ARRAYSEGV,GINPT(42)=N67,7,36
        ARRAYSEGV,F(42)=N67,43,16
$       BIT 60 IS EMPTY
$
        ARRAY,GATE(42)=N109      $$ N109 TO N150
$
        SEGMENT,LASTV=N41,1   $$  LASTV(I) MEANS THE LAST
$       VALUE PLOTTED FOR I-TH GRAPH
$
        SEGMENT,ORGLSTX=N42,10   $$ 18 ELEMENTS; 3 WORDS
        SEGMENT,ORGLSTY=N45,10   $$ LOCATION OF ORIGINS OF CHARTS
$       N42 TO N47
$
        TOUCHX=N48  $$ =ZTOUCHX.  ZTOUCHX/Y ARE COPIED TO TOUCHX/Y
$       IN ORDER TO ADJUST THE DIFFERENCE BETWEEN TOUCH-INPUT AND KEY-INPUT
        TOUCHY=N49  $$ =ZTOUCHY
        TOUCHV=N50  $$ REGION NUMBER OF TOUCH INPUT
$
        CHARTN=N51  $$ CURRENT CHART NUMBER
$
        CMMDCHF=N53  $$ COMMAND CHANGE FLAG.  ↑THIS IS 1
$       WHEN COMMAND IS TOUCHED WHILE PROCESSING OTHER COMMAND.
$
        KEYINPT=N54  $$ KEYINPT=1 MEANS KEY-INPUT
$
        SEGMENTF,TEMP0=N57,1,16  $$ TEMPLATE TO ACCESS ↑,F↑,
        SEGMENT,TEMPF=N57,1
        SEGMENTF,TEMPG=N58,1,36 $$ TEMPLATE TO ACCESS ↑,GINPT↑,
        SEGMENT,TEMP=N58,6
$
        R=N59 $$ ROW NUMBER OF LOCATION
$       THE RESULT OF TOUCH INPUT IS STORED IN ↑,R↑,
        C=N60 $$ COLUMN NUMBER OF LOCATION
        RC=N61 $$ SEQUENCE NUMBER OF LOCATION
$
$     CONSTANTS
DEFINE  MAXG=42
        INPCODE=7
        MAXFAN=6
```

```
54          DELAY=1  $$ DELAY TIME FOR ONE GATE. NOW CONSTANT.
55          MAXD=6 $$ MAXIMUM NUMBER OF CHARTS TO BE DISPLAYED
56  $
57          MSWHERE=405  $$ START LOCATION OF MESSAGE
58  $       LINES 4 AND 5 ARE USED FOR MESSAGES.
59          MSLNGTH=60  $$ MESSAGE MAXIMUM LENGTH
60  $
61          HEIGHT=12  $$ HEIGHT OF GRAPH
62          MAXT=27 $$ LENGTH OF CHART
63          TAU=8  $$ UNIT LENGTH OF TIME AXIS
```

```
64  $ MAJOR CHANGE. SUPPRESS KEY-INPUT FACILITY. BUT IT COMES
65  $  INTO EFFECT ONLY BY  TOGGLING LINES FROM ***...* TO
66  $  ===...= DOWN BELOW.

67  UNIT    MAIN0
68  BACK1    MAIN0
69  ERASE
70  DO      SCREEN
71  CALC    ALL↑0  $$ CLEARS ALL THE STUDENT VARIABLES
72  CALC    CHARTN↑1  $$ INITIALIZE CURRENT CHART NUMBER
73  CALC    CMMDCHF↑0  $$ INITIALIZE COMMAND CHANGE FLAG
74  CALC    ORGLSTX(1)↑ORGLSTX(2)↑ORGLSTX(3)↑0
75  CALC    ORGLSTX(4)↑ORGLSTX(5)↑ORGLSTX(6)↑256
76  CALC    ORGLSTY(1)↑ORGLSTY(4)↑32
77  CALC    ORGLSTY(2)↑ORGLSTY(5)↑16
78  CALC    ORGLSTY(3)↑ORGLSTY(6)↑0
79  $
80  CALC    KEYINPT↑0
81  $ **********
82  *DO      PRPMSG
83  *WRITE   ↑PRESS ↑,K↑, IF KEY INPUT, OTHERWISE PRESS ↑N↑E↑X↑T.
84  *ARROW   WHERE↑2
85  *LONG    1
86  *SPECS   NOOKNO
87  *ANSWER  K
88  *CALC    KEYINPT↑1
89  *OK
90  *CALC    KEYINPT↑0
91  *ENDARROW
92  $ ==========
93  GOTO    MAIN
94  $

95  UNIT    MAIN
96  INHIBIT ERASE
97  BACK1   MAIN0
98  BACK    MAIN
99  IF      CMMDCHF=1  $$ COMMAND CHANGE FLAG IS 1 WHEN
100 $       COMMAND IS TOUCHED WHILE PROCESSING OTHER COMMANDS.
101 .       CALC    CMMDCHF↑0
102 .       BRANCH  1CMMDCH
103 ENDIF
104 1UMAIN
```

```
105  DO       PRPMSG
106  WRITE    ↑WHICH COMMAND ↑/
107  DO       GETCMLN
108  ICMMDCH
109  IF       TOUCHV¬<7
110  .        DO       GATES
111  ELSEIF   TOUCHV=8
112  .        DO       PRPMSG
113  .        WRITE    ↑SORRY, ↑,REMOVE GATE↑, IS NOT IMPLEMENTED.
114  .        PAUSE    2
115  ELSEIF   TOUCHV=9
116  .        DO       CONNECT
117  ELSEIF   TOUCHV=10
118  .        DO       PRPMSG
119  .        WRITE    ↑SORRY, ↑,DISCONNECT↑, IS NOT IMPLEMENTED.
120  .        PAUSE    2
121  ELSEIF   TOUCHV=11  $$ INPUT VALUE SEQUENCE FOR INPUT TERMINALS
122  .        DO       INPVSEQ
123  ELSEIF   TOUCHV=12  $$ SET GATE INITIAL VALUE
124  .        DO       GTINITL
125  ELSEIF   TOUCHV=13
126  .        DO       CALCUL
127  .        DO       CHART
128  ELSEIF   TOUCHV=14  $$ ERASE CHARTS
129  .        AT       0,32
130  .        ERASE    64,3
131  .        CALC     CHARTN←1
132  ELSEIF   TOUCHV=15  $$ REFRESH SCREEN
133  .        JUMP     MAIN0
134  ENDIF
135  BRANCH   1UMAIN
136  $$$$$

137  UNIT     MAINEND
```

```
138  UNIT     SCREEN  $$ LEVEL 1
139  $  SETS UP INITIAL SCREEN
140  DEFINE   SCRI=N1
141           SCRR=N2
142           SCRC=N3
143  DEFINE   UNT=32
144  RORIGIN  0,464
145  RBOX     0,0;511,47;-2
146  DUTO     1BOXEND,SCRI←1,15
147  RDRAW    UNT≠SCR1,47;UNT≠SCRI,0
148  1BOXEND
149  *
150  RAT      7,12
151  WRITE    ≥AND
152  RAT      UNT+6,12
153  WRITE    ≥0↑RR
154  RAT      2UNT+10,12
155  WRITE    ≥IP
```

```
156   RAT       3UNT+6,12
157   WRITE     ≥AN1
158   RAT       4UNT+6,12
159   WRITE     ≥O↑RQ
160   RAT       5UNT+6,12
161   WRITE     ≥234
162   RAT       6UNT+9,12
163   WRITE     ≥BC
164   *
165   RAT       4.28
166   WRITE     AND
167   RAT       UNT+8,28
168   WRITE     OR
169   RAT       2UNT+4,28
170   WRITE     NOT
171   RAT       3UNT+1,25
172   WRITE     NAND
173   RAT       4UNT+4,28
174   WRITE     NOR
175   RAT       5UNT+4,28
176   WRITE     XOR
177   RAT       6UNT+1,26
178   WRITE     INPT
179   †
180   RAT       7UNT+4,28
181   WRITE     RMV
182   RAT       7UNT+1,9
183   WRITE     GATE
184   ‡
185   RAT       8UNT+1,17
186   WRITE     CNCT
187   §
188   RAT       9UNT+4,28
189   WRITE     DIS
190   RAT       9UNT+1,9
191   WRITE     CNCT
192   ¶
193   RAT       10UNT+6,34
194   WRITE     INPT
195   RAT       10UNT+4,16
196   WRITE     VAL
197   RAT       10UNT+4,2
198   WRITE     SEQ
199   ‖
200   RAT       11UNT+1,34
201   WRITE     GATE
202   RAT       11UNT+1,16
203   WRITE     INTL
204   RAT       11UNT+3,2
205   WRITE     VAL
206   #
207   RAT       12UNT+1,28
208   WRITE     SHOW
209   RAT       12UNT+1,9
210   WRITE     CHRT
211   ¤
212   RAT       13UNT+1,28
```

```
213  WRITE    ERAS
214  RAT      13UNT+1,9
215  WRITE    CHRT
216  $
217  RAT      14UNT+8,34
218  WRITE    RE
219  RAT      14UNT+1,18
220  WRITE    FRSH
221  RAT      14UNT+1,2
222  WRITE    SCRN
223  $
224  DOTO     1ROW,SCRR+1,6,1
225  DOTO     1COL,SCRC+1,7,1
226  AT       ((SCRC#64-6)-18),((448-64#SCRR)+8)
227  WRITE    -0S,SCRR-1-0S,SCRC-1
228  1COL
229  1ROW
230  $$$$$

231  UNIT     GATES  $$ LEVEL 2
232  $    SETS TYPE AND PLACES A GATE AT THE SPECIFIED POSITION
233  $    CONTROL IS PASSED WITH TOUCHV=1 THROUGH 7
234  $    IF COMMAND IS TOUCHED, GOES BACK TO THE MAIN ROUTINE
235  DEFINE   GKIND=N11
236  CALC     GKIND+TOUCHV
237  0UGATE
238  DO       PRPMSG
239  WRITE    ↑,
240  WRITEC   GKIND,,,AND,OR,NOT,NAND,NOR,XOR,INPT
241  WRITE    ↑, AT WHICH LOCATION↑/
242  1UGATE
243  DO       GETLCTN
244  $     WHEN NETWORK IS TOUCHED
245  IF       NOT(TYPE(RC)=0)
246  .        DO       PRPMSG
247  .        WRITE    ↑THERE IS ALREADY A GATE OR INPUT THERE ↑.
248  .        PAUSE    2
249  .        BRANCH   0UGATE
250  ELSE
251  .        CALC     TYPE(RC)+GKIND
252  .        AT       TOUCHX-8,TOUCHY-8
253  .        ERASE    2
254  .        AT       TOUCHX-19,TOUCHY-8
255  .        WRITE
256  .        WRITEC   GKIND,,,≥AND≥,≥0↑RR≥,≥IP≥,≥AN1≥,≥0↑RQ≥,≥234≥,≥BC
257  .        BRANCH   1UGATE
258  ENDIF
259  1GTEND
```

```
260  UNIT     CONNECT  $$ LEVEL 3
261  DEFINE   R0=N21 $$ ROW NUMBER OF ORIGINATING GATE
262           C0=N22 $$ COLUMN NUMBER OF ORIGINATING GATE
263           PC0=N23 $$ SEQUENCE NUMBER OF ORIGINATING GATE
```

```
264             ZTX=N24 $$ X-COORDINATE OF ORIGINATING GATE
265             ZTY=N25 $$ Y
266     DEFINE  HPATHN=N28  $$ HORIZONTAL PATH NUMBER
267             INPUTN=N27  $$ INPUT LINE NUMBER FOR TARGET GATE
268     0UCON
269     DO      PRPMSG
270     WRITE   ↑CONNECT FROM WHICH GATE OR INPUT ↑/
271     DO      GETLCTN
272     IF      TYPE(RC)=0
273     .       DO      PRPMSG
274     .       WRITE   ↑THERE IS NO GATE THERE ↑.
275     .       PAUSE   2
276     .       BRANCH  0UCON
277     ENDIF
278     CALC    ZTX→TOUCHX
279     CALC    ZTY→TOUCHY
280     CALC    R0→R
281     CALC    C0→C
282     CALC    RC0→RC
283     *
284     1UCON
285     DO      PRPMSG
286     WRITE   ↑FROM
287     WRITEC  TYPE(RC0)-7, GATE, INPUT
288     WRITE    ¬0S,R0¬1¬0S,C0¬1 TO WHICH GATE ↑/
289     DO      FLASH(R0,C0,TYPE(RC0))  $$ FLASH THE SOURCE GATE
290     DO      GETLCTN
291     IF      TYPE(RC)=0 $OR$ TYPE(RC)>6
292     .       DO      PRPMSG
293     .       WRITE   ↑THERE IS NO GATE THERE ↑.
294     .       PAUSE   2
295     .       BRANCH  1UCON
296     ENDIF
297     CALC    TEMPG→GINPT(RC)
298     IF      (TEMP(1)=RC0)$OR$(TEMP(2)=RC0)$OR$(TEMP(3)=RC0)$OR$
                (TEMP(4)=RC0)$OR$(TEMP(5)=RC0)$OR$(TEMP(6)=RC0)
299     .       DO      PRPMSG
300     .       WRITE   ↑THIS CONNECTION HAS ALREADY BEEN DRAWN↑.
301     .       PAUSE   2
302     .       BRANCH  0UCON
303     ENDIF
304     2UCON
305     IF      R>RC  $$ ORIGINATING GATE IS UPPER THAN DESTINATION
306     $       SELECT A PAIR OF INPUT AND PATH FROM THE UPPER
307     $       THREE INPUTS AND THE UPPER EIGHT PATHS
308     .       DO      UPPER(INPUTN,HPATHN) $$ FIRST PRIORITY
309     .       IF      HPATHN=-1
310     $               SELECT A PAIR OF INPUT AND PATH FROM THE
311     $               LOWER THREE INPUTS AND EIGHT PATHS
312     .       .       DO      LOWER(INPUTN,HPATHN) $$ SECOND PRIORITY
313     .       .       IF      HPATHN=-1
314     .       .       .       DO      PRPMSG
315     .       .       .       WRITE   ↑NO AVAILABLE INPUT OR PATH.
316     .       .       .       PAUSE   2
317     .       .       .       BRANCH  0UCON
318     .       .       ENDIF
319     .       ENDIF
320     ELSE
```

```
$  ORIGINATING GATE IS LOWER
.       DO      LOWER(INPUTN,HPATHN)
.       IF      HPATHN=-1
.       .       DO      UPPER(INPUTN,HPATHN)
.       .       IF      HPATHN=-1
.       .       .       DO      PRAMSG
.       .       .       WRITE   ↑NO AVAILABLE INPUT OR PATH.
.       .       .       PAUSE   2
.       .       .       BRANCH  OUCON
.       .       ENDIF
.       ENDIF
ENDIF
$ RECORD THE SELECTED INPUT LINE.
CALC    TEMPG→GINPT(RC)
CALC    TEMP(INPUTN)→RC0
CALC    GINPT(RC)→TEMPG
$  RECORD THE SELECTED PATH
DO      RCRDPTH(HPATHN)
$ DRAW CONNECTION
DO      DRAWCNN
BRANCH  OUCON
```

```
UNIT    LOWER(INPUTN,HPATHN)  $$ LEVEL 2
$  FINDS AN INPUT LINE OUT OF THE UPPER THREE AND
$  A HORIZONTAL PATH OUT OF THE UPPER EIGHT.
DO      FINDPTH(1,HPATHN)
DO      FINDINP(1,INPUTN)
IF      HPATHN=-1 $OR$ INPUTN=-1
.       CALC    INPUTN→-1
.       CALC    HPATHN→-1
ENDIF
$$$$$

UNIT    UPPER(INPUTN,HPATHN)  $$ LEVEL 2
DO      FINDPTH(9,HPATHN)
DO      FINDINP(4,INPUTN)
IF      INPUTN=-1 $OR$ HPATHN=-1
.       CALC    INPUTN→-1
.       CALC    HPATHN→-1
ENDIF
$$$$$

UNIT    FINDINP(N1,INPUTN)  $$ LEVEL 1
$ FIND AN AVAILABLE INPUT TO GATE(RC)
$     FROM THE LOWER THREE WHEN ↑,LORU↑, IS 1
$     FROM THE UPPER THREE WHEN ↑,LORU↑, IS 4
DEFINE  LORU=N1  $$ 1 OR 4.  1 MEANS LOWER, 4 MEANS UPPER.
CALC    TEMPG→GINPT(RC)
IF      TEMP(LORU)=0
.       CALC    INPUTN→LORU
ELSEIF  TEMP(LORU+1)=0
.       CALC    INPUTN→LORU+1
ELSEIF  TEMP(LORU+2)=0
```

```
371   .       CALC    INPUTN←LORU+2
372   ELSE
373   .       CALC    INPUTN←-1
374   ENDIF
375   $$$$$$$$$$$

376   UNIT    FINDPTH(N1,HPATHN)  $$ LEVEL 1
377   DEFINE  STARTC=N1
378   $ FIND AN AVAILABLE PATH BETWEEN COLUMN-C AND COLUMN-C0
379   $  STARTC=1 OR 9.  1 MEANS LOWER, 9 MEANS UPPER.
380   DEFINE  FPRC=N2,FPRC1=N3,FPRC2=N4
381           HPATHX=N5
382   IF      C<C0
383   .       CALC    FPRC1←RC
384   .       CALC    FPRC2←RC+C0-C
385   ELSE
386   .       CALC    FPRC1←RC+C0-C
387   .       CALC    FPRC2←RC
388   ENDIF
389   DOTO    1FPLOOP,HPATHX←STARTC,STARTC+7
390   DOTO    2FPLOOP,FPRC←FPRC1,FPRC2
391   CALC    TEMP0←F(FPRC)
392   IF      TEMPF(HPATHX)¬=0
393   .       BRANCH  1FPNG
394   ENDIF
395   2FPLOOP
396   CALC    HPATHN←HPATHX
397   BRANCH  1FPOUT
398   1FPNG
399   1FPLOOP
400   CALC    HPATHN←-1
401   1FPOUT
402   $$$$$$$$$$$
```

```
403   UNIT    RCRDPTH(HPATHN)
404   $ SETS ↑,HPATHN↑,-TH BIT OF F(RC0) THROUGH F(RC) TO RECORD THAT
      ↑,HPATHN↑,-TH PATH IS OCCUPIED.
405   DEFINE  RCRCOL=N1,RCRCOL1=N2,RCRCOL2=N3
406   IF      C<C0
407   .       CALC    RCRCOL1←RC
408   .       CALC    RCRCOL2←RC+C0-C
409   ELSE
410   .       CALC    RCRCOL1←RC+C0-C
411   .       CALC    RCRCOL2←RC
412   ENDIF
413   DOTO    1RCRL,RCRCOL←RCRCOL1,RCRCOL2
414   CALC    TEMP0←F(RCRCOL)
415   CALC    TEMPF(HPATHN)←1
416   CALC    F(RCRCOL)←TEMP0
417   1RCRL
418   $$$$$$$$$$$

419   UNIT    DRAWCNN  $$ LEVEL 1
420   $ DRAW CONNECTION
```

```
421  DRAW     ZTX-2,ZTY;ZTX+6+R0#3,ZTY
422  IF       HPATHN¬<8  $$ LOWER PATH OF GATE(RC)
423  .        DRAW     ;WHEREX,TOUCHY-(3#HPATHN+7)
424  .        DRAW     ;TOUCHX-3#INPUTN-10,WHEREY
425  .        DRAW     ;WHEREX,TOUCHY+2#INPUTN-8
426  ELSE
427  .        DRAW     ;WHEREX,TOUCHY+3#HPATHN-18
428  .        DRAW     ;TOUCHX-3#(INPUTN-3)-10,WHEREY
429  .        DRAW     ;WHEREX,TOUCHY-2#INPUTN+12
430  ENDIF
431  DRAW     ;TOUCHX-9,WHEREY
432  $
433  $  WRITES DOTS AT THE INTERSECTIONS OF THE FIRST, THE SECOND,
        AND THE THIRD LINE SEGMENTS.
434  AT       ZTX+3+R0#3,ZTY-4
435  WRITE    .
436  IF       HPATHN¬<8
437  .        AT       ZTX+3+R0#3,TOUCHY-3#HPATHN-11
438  ELSE
439  .        AT       ZTX+3+R0#3,TOUCHY+3#HPATHN-22
440  ENDIF
441  WRITE    .
442  $$$$$$$$$$

443  UNIT     FLASH(N1,N2,N3)  $$ LEVEL 1
444  DEFINE   FLR=N1,FLC=N2,FLX=N3
445  MODE     ERASE
446  AT       64FLC-27,464-64FLR-8
447  WRITEC   FLX,,,≥AND≥,≥0↑RR≥,≥IP≥,≥AN1≥,≥0↑RQ≥,≥234≥,≥BC
448  MODE     WRITE
449  AT       64FLC-27,464-64FLR-8
450  WRITEC   FLX,,,≥AND≥,≥0↑RR≥,≥IP≥,≥AN1≥,≥0↑RQ≥,≥234≥,≥BC
451  MODE     ERASE
452  AT       64FLC-27,464-64FLR-8
453  WRITEC   FLX,,,≥AND≥,≥0↑RR≥,≥IP≥,≥AN1≥,≥0↑RQ≥,≥234≥,≥BC
454  MODE     WRITE
455  AT       64FLC-27,464-64FLR-8
456  WRITEC   FLX,,,≥AND≥,≥0↑RR≥,≥IP≥,≥AN1≥,≥0↑RQ≥,≥234≥,≥BC
```

```
457  UNIT     CALCUL  $$ LEVEL 1 AND 2
458  DEFINE   ARRAY,GINPTL(6)=N11  $$ N11 TO N16, 6=MAXFAN
459           IT=N1,IG=N2,II=N3,CGINPTN=N4,CTYPE=N5,RESULT=N6
460           SEGMENT,CGINPT=N4,6
461  DO       PRPMSG
462  WRITE    ↑NOW CALCULATING.
463  $   CLEARS GATE EXCEPT THE FIRST BIT AND EXCEPT INPUT TERMINALS
464  DOTO     9END,II↑1,MAXG
465  IF       TYPE(II)¬>1 $AND$ TYPE(II)¬<6
466  .        CALC     GATE(II)↑GATE(II) $MASK$ (01 $CLS$ 59)
467  ENDIF
468  9END
469  DOTO     1END,IT↑1+DELAY,MAXT
470  DOTO     2END,IG↑1,MAXG
471  IF       TYPE(IG)=0 $OR$ TYPE(IG)=INPCODE
```

```
472  .       BRANCH  2END
473  ENDIF
474  CALC    CGINPTW←GINPT(IG)
475  CALC    CGINPTW←CGINPTW $CLS$ 24
476  CALC    CTYPE←TYPE(IG)
477  IF      CTYPE=1 $OR$ CTYPE=4        $$ AND, NAND
478  .       CALC    GINPTL←1
479  ELSE
480  .       CALC    GINPTL←0
481  ENDIF
482  DOTO    3END,II←1,MAXFAN
483  IF      CGINPT(II)=0
484  .       BRANCH  3END
485  ENDIF
486  CALC    GINPTL(II)←(GATE(CGINPT(II)) $CLS$ (IT-DELAY)) $MASK$ 01
487  3END
488  3ENDL
489  IF      CTYPE=1                     $$ AND
490  .       IF      GINPT(IG)=0  $$ NO INPUTS
491  .       .       CALC    RESULT←0
492  .       ELSE
493  .       .       CALC    RESULT←↑PROD(GINPTL)
494  .       ENDIF
495  ELSEIF  CTYPE=2                     $$ OR
496  .       IF      ↑SUM(GINPTL)=0
497  .       .       CALC    RESULT←0
498  .       ELSE
499  .       .       CALC    RESULT←1
500  .       ENDIF
501  ELSEIF  CTYPE=3                     $$ NOT
502  .       CALC    RESULT←0
503  .       DOTO    4END,II←1,MAXFAN
504  .       CALC    RESULT←RESULT+CGINPT(II)≠GINPTL(II)
505  .       4END
506  .       IF      RESULT¬=0
507  .       .       CALC    RESULT←0
508  .       ELSE
509  .       .       CALC    RESULT←1
510  .       ENDIF
511  ELSEIF  CTYPE=4                     $$ NAND
512  .       IF      GINPT(IG)=0  $$ NO INPUTS
513  .       .       CALC    RESULT←1
514  .       ELSE
515  .       .       CALC    RESULT←↑PROD(GINPTL) $DIFF$ 01
516  .       ENDIF
517  ELSEIF  CTYPE=5                     $$ NOR
518  .       IF      ↑SUM(GINPTL)=0
519  .       .       CALC    RESULT←1
520  .       ELSE
521  .       .       CALC    RESULT←0
522  .       ENDIF
523  ELSEIF  CTYPE=6                     $$ EOR
524  .       CALC    RESULT←GINPTL(1) $DIFF$ GINPTL(2)  $DIFF$ GINPTL(3)
                     $DIFF$ GINPTL(4) $DIFF$ GINPTL(5) $DIFF$ GINPTL(6)
525  ENDIF
526  CALC    GATE(IG)←GATE(IG) $UNION$ (RESULT $CLS$ 60-IT)
527  2END
528  1END
```

```
529  UNIT     CHART  $$ LEVEL 3
530  DEFINE   VARGTN=N21
531  1NEXTGR
532  DO       SELECT(VARGTN)  $$ RESULT IS THE SPECIFIED GATE NUMBER
533  DO       AXES(VARGTN,ORGLSTX(CHARTN),CRGLSTY(CHARTN))
534  DO       PLOT(VARGTN,ORGLSTX(CHARTN),CRGLSTY(CHARTN))
535  DO       INCCHRT
536  BRANCH   1NEXTGR
537  $$$$$$$$$$

538  UNIT     SELECT(VARGTN)  $$ LEVEL 2
539  1AGAIN
540  DO       PRPMSG
541  WRITE    ↑WHICH GATE OR INPUT↑/
542  DO       GETLCTN
543  IF       TYPE(RC)¬>1 $AND$ TYPE(RC)¬<7
544  .        CALC     VARGTN←RC
545  ELSE
546  .        DO       PRPMSG
547  .        WRITE    ↑THERE IS NO GATE OR INPUT.
548  .        PAUSE    2
549  .        BRANCH   1AGAIN
550  ENDIF
551  $$$$$$$$$$

552  UNIT     PLOT(N11,N12,N13)
553  DEFINE   PLVARGN=N11,PLLOCX=N12,PLLOCY=N13
554           IIP=N14   $$ CORRESPONDS TO TIME IIP-1 TO IIP
555  RORIGIN PLLOCX+28,PLLOCY+2
556  DOTO     2END,IIP←1,MAXT
557  DO       DRAW1((GATE(PLVARGN) $CLS$ IIP)  $MASK$ 01, IIP)
558  2END
559  $$$$$

560  UNIT     INPVSEQ  $$ LEVEL 2
561  $  ASSIGNS AN INPUT VALUE SEQUENCE TO AN INPUT TERMINAL.
562  DEFINE   SVI=N12,VARV=N13
563  1AGAIN
564  DO       PRPMSG
565  WRITE    ↑WHICH INPUT↑/
566  DO       GETLCTN
567  IF       TYPE(RC)¬=INPCODE
568  .        DO       PRPMSG
569  .        WRITE    ↑THAT IS NOT AN INPUT.
570  .        PAUSE    2
571  .        BRANCH   1AGAIN
572  ENDIF
573  CALC     GATE(RC)←0
```

```
574  DO       AXES(RC,ORGLSTX(CHARTN),ORGLSTY(CHARTN))
575  DO       FRPMSG
576  WRITE    ↑PRESS 0,1,Z(CONSTANT 0) OR C(CONSTANT 1).
577  DOTO     1END,SVI←1,MAXT
578  1WRNGV
579  ARROW    MSWHERE+41
580  LONG     1
581  SPECS    NOOKNO
582  ANSWER   0
583  CALC     VARV←0
584  ANSWER   1
585  CALC     VARV←1
586  ANSWER   Z
587  CALC     VARV←2
588  ANSWER   C
589  CALC     VARV←3
590  OK
591  CALC     VARV←-1
592  ENDARROW
593  AT       MSWHERE+43
594  ERASE    1
595  IF       VARV<0
596  .        BRANCH   1WRNGV
597  ELSEIF   VARV¬<1
598  .        DO       SETVAL
599  ELSE
600  .        CALC     VARV←VARV-2
601  .        GOTO     2END,SVI←SVI,MAXT
602  .        DO       SETVAL
603  .        2END
604  .        BRANCH   9END
605  ENDIF
606  1END
607  9END
608  DO       INCCHRT  $$ INCREASE CHARTN BY ONE
609  BRANCH   1AGAIN
610  $

611  UNIT     SETVAL   $$ LEVEL 2
612  CALC     GATE(RC)←GATE(RC) $UNION$ (VARV $CLS$ 60-SVI)
613  DO       DRAW1(VARV,SVI)
614  $$$$$$$$$$

615  UNIT     GTINITL  $$ LEVEL 2
616  $    SETS INITIAL VALUES OF GATES
617  DEFINE   SII=N11
618  DOTO     3END,SII←1,MAXG
619  IF       TYPE(SII)¬>1 $AND$ TYPE(SII)¬<6
620  .        CALC     GATE(SII)←0
621  ENDIF
622  3END
623  GNEXT
624  DO       FRPMSG
625  WRITE    ↑TOUCH GATES WITH INITIAL VALUE 1.
626  1NEXT
627  DO       GETLCTN
628  IF       TYPE(RC)=0 $OR$ TYPE(RC)¬>7
```

```
629  .        DO       PRPMSG
630  .        WRITE    ↑THAT IS NOT A GATE.
631  .        PAUSE    2
632  .        BRANCH   0NEXT
633  ENDIF
634  CALC     GATE(RC)↠01 $CLS$ 59
635  AT       MSWHERE↑35
636  ERASE    30-(MSWHERE-INT(MSWHERE/100)≠100)
637  AT       MSWHERE↑35
638  WRITE    G¬0S,R¬1¬0S,C¬1 OK
639  BRANCH   1NEXT
640  1SETEND
641  $$$$$
```

```
642  UNIT     AXES(N2,N3,N4)  $$ LEVEL 1
643  DEFINE   AXIGN=N2,LOCX=N3,LOCY=N4
644  DEFINE   AXI=N5,AXROW=N6,AXCOL=N7
645  $
646  $              AXIGN ↑; LOCATION OF NETWORK, INPUT VARIABLE OR GATE
647  $              LOCX,LOCY ↑; LOCATION IN FINE GRID
648  $              HEIGHT AND TAU ARE DEFINED IN THE MAIN UNIT
649  RORIGIN LOCX,LOCY
650  RAT      0,0
651  ERASE    32
652  RAT      0,1
653  CALC     AXROW↠INT((AXIGN-1)/7)↑1
654  CALC     AXCOL↠AXIGN-AXROW≠7↑7
655  IF       TYPE(AXIGN)¬<6
656  .        WRITE    G
657  ELSE
658  .        WRITE    I
659  ENDIF
660  WRITE    ¬0S,AXROW¬1¬0S,AXCOL¬1
661  RORIGIN LOCX↑28,LOCY↑2
662  RDRAW    0,0;MAXT≠TAU,0;SKIP;0,HEIGHT;0,0
663  DOTO     1END,AXI↠1,MAXT
664  RDRAW    AXI≠TAU,2;AXI≠TAU,0
665  1END
666  $$$$$

667  UNIT     DRAW1(N2,N3)  $$ LEVEL 1
668  $   CHARTN SPECIFIES THE CURRENT NUMBER OF THE CHART.
669  DEFINE   DRV=N2  $$ VALUE TO BE PLOTTED
670           DRT=N3  $$ TIME SEGMENT ( DRT-1<T<DRT )
671           DRX=N4
672           DRY=N5
673  $         LASTV IS DEFINED IN THE MAIN UNIT
674  CALC     DRX↠TAU≠(DRT-1)
675  CALC     DRY↠HEIGHT≠DRV
676  IF       DRV ¬= LASTV(CHARTN) $AND$ DRT ¬= 1
677  .        RDRAW    DRX,HEIGHT-DRY;DRX,DRY;SKIP;DRX↑1,HEIGHT-DRY;DRX↑1,DRY
678  ENDIF
679  RDRAW    DRX,DRY;DRX↑TAU,DRY;SKIP;DRX,DRY↑1;DRX↑TAU,DRY↑1
```

```
680  CALC     LASTV(CHARTN)→DRV
681  $$$$$$$$$$

682  UNIT     FRPMSG
683  AT       MSWHERE
684  ERASE    MSLNGTH,2
685  AT       MSWHERE
686  $$$$$

687  UNIT     FRPMSG2
688  AT       MSWHERE+100
689  ERASE    MSLNGTH
690  AT       MSWHERE+100
691  $$$$$

692  UNIT     INCCHRT  $$ INCREASE CHART NUMBER BY ONE
693  CALC     CHARTN→CHARTN+1
694  IF       CHARTN=7
695  .        CALC     CHARTN→1
696  ENDIF
697  $$$$$
```

```
698  UNIT     GETLCTN  $$ LEVEL 1
699  $   GET F, C, RC OF LOCATION
700  DEFINE   GTWHERE=N9
701  DO       GETCMLN
702  IF       TOUCHV¬>1 $AND$ TOUCHV¬<17
703  .        CALC     CMMDCHF→1
704  .        JUMP     MAIN
705  ENDIF
706  *AT       3102
707  *ERASE    50
708  *AT       3102
709  *WRITE    RC=¬0S,RC¬1,R=¬0S,R¬1,C=¬0S,C¬1
710  $$$$$

711  UNIT     GETCMLN  $$ GET COMMAND OR LOCATION
712  $   GET TOUCHV OF COMMAND OR LOCATION.
713  $
714  IF       KEYINPT=1
715  .        DO       GETKEY
716  ELSE
717  .        DO       GETTCH
718  ENDIF
719  *AT       3110
720  *ERASE    25
721  *AT       3110
722  *WRITE    TOUCHV OR KEYIN=¬0S,TOUCHV-1
723  $$$$$

724  UNIT     GETKEY
725  CALC     GTWHERE→MSWHERE+100
726  ARROW    GTWHERE
```

```
727  MATCH    TOUCHV,BACK,AND,OR,NOT,NAND,NOR,XOR,INPT,
728           (RMV GATE,RMVGATE,RMV),
729           CNCT,(DIS,DIS CNCT,DISCNCT),
730           (INPT VAL,INPTVAL),
731           (GATE INTL VAL,GATEINTLVAL,INTL VAL,INTLVAL,GATE INTL,GATEINTL),
732           (SHOW CHRT,SHOWCHRT,SHOW CHART,CHART,SHOWCHART),
733           (ERAS GATE AREA,ERASGATEAREA,ERAS GATE),,,,
734           11,12,13,14,15,16,17,
735           21,22,23,24,25,26,27,
736           31,32,33,34,35,36,37,
737           41,42,43,44,45,46,47,
738           51,52,53,54,55,56,57,
739           61,62,63,64,65,66,67,BACK1,STOP1
740  NO
741  PAUSE
742  ENDARROW
743  IF       TOUCHV=0   $$ ↑B↑A↑C↑K
744  .        JUMP     MAIN
745  ELSEIF   TOUCHV=60  $$ ↑B↑A↑C↑K1
746  .        JUMP     MAIN0
747  ELSEIF   TOUCHV=61  $$ ↑S↑T↑O↑P1
748  .        JUMP     MAINEND
749  ENDIF
750  DO       MODIFY
751  CALC     TOUCHX←64≠C-16
752  CALC     TOUCHY←464-64≠R
753  $$$$$
```

```
754  UNIT     GETTCH
755  $  ACCEPT A TOUCH PANEL INPUT↑; TOUCHV=0   ↑; ↑B↑A↑C↑K
756  $                                1¬<TOUCHV¬<17 ↑; COMMAND
757  $                                18¬<TOUCHV¬<59 ↑; NETWORK POSITION
758  $                                60 ↑; ↑B↑A↑C↑K1
759  ENABLE   TOUCH
760  PAUSE
761  TUIFL
762  KEYTYPE TOUCHV,BACK,T(401,4,3),T(405,4,3),T(409,4,3),T(413,4,3),
763           T(417,4,3),T(421,4,3),T(425,4,3),T(429,4,3),
764           T(433,4,3),T(437,4,3),T(441,4,3),T(445,4,3),
765           T(449,4,3),T(453,4,3),T(457,4,3),T(461,4,3),
766           T(465,4,3),T(805,4,2),T(813,4,2),T(821,4,2),   $$ ↑/↑/↑/
767           T(829,4,2),T(837,4,2),T(845,4,2),T(853,4,2),
768           T(1205,4,2),T(1213,4,2),T(1221,4,2),T(1229,4,2),
769           T(1237,4,2),T(1245,4,2),T(1253,4,2),T(1605,4,2),
770           T(1613,4,2),T(1621,4,2),T(1629,4,2),T(1637,4,2),
771           T(1645,4,2),T(1653,4,2),T(2005,4,2),T(2013,4,2),
772           T(2021,4,2),T(2029,4,2),T(2037,4,2),T(2045,4,2),
773           T(2053,4,2),T(2405,4,2),T(2413,4,2),T(2421,4,2),
774           T(2429,4,2),T(2437,4,2),T(2445,4,2),T(2453,4,2),
775           T(2605,4,2),T(2813,4,2),T(2821,4,2),T(2829,4,2),
776           T(2837,4,2),T(2845,4,2),T(2853,4,2),BACK1,STOP1
777  *
778  IF       TOUCHV=-1
```

```
779  .       DO      PRPMSG2
780  .       WRITE   ↑TRY AGAIN PLEASE.
781  .       PAUSE
782  $ ↑WHY THE PROGRAM GOES INTO AN INFINITE LOOP.  ↑/↑/↑/↑/↑/↑/↑/↑/↑/↑/↑/
783  $ WHEN A DURATION OF PAUSE IS SPECIFIED↑/        ↑/↑/↑/↑/↑/↑/↑/↑/↑/↑/↑/
784  .       BRANCH  1UIFL
785  ELSEIF  TOUCHV=0  $$ ↑B↑A↑C↑K
786  .       JUMP    MAIN
787  ELSEIF  TOUCHV=60  $$ ↑B↑A↑C↑K1
788  .       JUMP    MAIN0
789  ELSEIF  TOUCHV=61  $$ ↑S↑T↑O↑P1
790  .       JUMP    MAINEND
791  ENDIF
792  DISABLE TOUCH
793  CALC    TOUCHX→ZTOUCHX
794  CALC    TOUCHY→ZTOUCHY
795  DO      MODIFY
796  $$$$$

797  UNIT    MODIFY
798  CALC    RC→TOUCHV-17
799  CALC    R→INT((RC-1)/7)+1
800  CALC    C→RC-7*(R-1)
801  $$$$$
802  CSTOP*
```

BIBLIOGRAPHIC DATA SHEET	1. Report No. UIUCDCS-R-79-979	2.	3. Recipient's Accession No.
4. Title and Subtitle A manual of SIMPL: A logic network simulation for PLATO			5. Report Date July 24, 1979 6.
7. Author(s) Sadahiro Isoda and Elyas Farzan-Kashani			8. Performing Organization Rept. No. UIUCDCS-R-79-979
9. Performing Organization Name and Address Department of Computer Science University of Illinois U-C Urbana, Illinois 61801			10. Project/Task/Work Unit No. 11. Contract/Grant No.
12. Sponsoring Organization Name and Address			13. Type of Report & Period Covered Technical 14.

15. Supplementary Notes

16. Abstracts

This is a user's manual of SIMPL, a logic network simulator for PLATO. SIMPL is a program which can interactively simulate logic networks, based on the computer-based education system, PLATO. SIMPL enables the user to construct any network consisting of logic gates (AND, OR, NOT, NAND, NOR, or XOR) and input terminals, to supply input value sequences to input terminals, to assign initial values to gates, and to show timing charts for any gates of the network.

SIMPL is quite easy to use even for a novice, because the user, following messages from SIMPL, cna use it almost by means of touching the screen.

17. Key Words and Document Analysis. 17a. Descriptors

computer-based education, interactive simulation, logic network, logic simulation, PLATO, terminal, touch panel

17b. Identifiers/Open-Ended Terms

17c. COSATI Field/Group

18. Availability Statement	19. Security Class (This Report)	21. No. of Pages
unlimited	UNCLASSIFIED	41
	20. Security Class (This Page) UNCLASSIFIED	22. Price

JUN 1 2 1980

FEB 20 1981

BOUND BY
HNM

UNIVERSITY OF ILLINOIS-URBANA
3 0112 045361299